

Amendments to the Claims

Please add or amend the claims to read as follows, and cancel without prejudice or disclaimer to resubmission in a divisional or continuation application claims indicated as cancelled:

1. (Previously presented) A method comprising:
during translation of a code block in a system comprising a hardware processor from a first format suitable for a first computing platform to a second format suitable for a second computing platform, inserting one or more instructions in said code block to detect whether execution of said code block results in a misaligned data access prior to execution of said code block;
storing a location of a first memory address in a tracking list for storing the location of memory addresses for which misaligned data access is detected if accessing the first memory address by a first instruction results in the misaligned data access;
adding one or more instructions to the first instruction;
checking the tracking list to determine if a location of a second memory address is identical to the location of the first memory address when a second instruction requires access to the second memory address; and
obviating the need to add instructions to the second instruction if the location of the second memory address accessed by the second instruction is identical to the location of the first memory address.
2. (Cancelled)
3. (Previously Presented) The method of claim 1 wherein detecting comprises inserting at least one instruction in said code block to detect a location of an instruction whose execution results in the misaligned data access.
4. (Previously Presented) The method of claim 1, wherein inserting one or more instructions in said code block comprises inserting at least one instruction in said code

- block to detect a location of an instruction whose execution results in the misaligned data access.
5. (currently amended) The method of claim 1, wherein ~~modifying~~ adding comprises adding to said code block an instruction to branch an execution of said code block to a code sequence whose execution handles the misaligned data access.
 6. (currently amended) The method of claim 1, wherein ~~modifying~~ comprises comprising modifying said code block to handle misaligned data access in a subsequent execution of said code block.
 7. (Original) The method of claim 1, further comprising translating said code block from said first format to said second format.
 8. (Previously Presented) The method of claim 1, wherein inserting one or more instructions in said code block further comprises inserting one or more instructions in said code block to detect whether execution of said code block results in the misaligned data access prior to execution of a code block translated from a format suitable for a 32-bit based computing platform to a format suitable for a 64-bit based computing platform.
 9. (Previously presented) An apparatus comprising:
a hardware processor to insert one or more instructions, during translation of a code block from a first format suitable for a first computing platform to a second format suitable for a second computing platform, in said code block to detect whether execution of said code block results in misaligned data access prior to execution of said code block, to store a location of a first memory address in a tracking list for storing the location of memory addresses for which misaligned data access is detected if accessing the first memory address by a first instruction results in the misaligned data access; to add one or more instructions to the first instruction; to check the tracking list to determine if a location of a second memory address is identical to the

location of the first memory address when a second instruction requires access to the second memory address; and to obviate the need to add instructions to the second instruction if the location of the second memory address accessed by the second instruction is identical to the location of the first memory address.

10. (Cancelled)
11. (Previously Presented) The apparatus of claim 9 wherein the processor is able to insert at least one instruction in said code block to detect a location of an instruction whose execution results in the misaligned data access.
12. (Cancelled).
13. (Original) The apparatus of claim 9, wherein the processor is able to add to said code block an instruction to branch an execution of said code block to a code sequence whose execution handles the misaligned data access.
14. (Original) The apparatus of claim 9, wherein the processor is able to modify said code block to handle misaligned data access in a subsequent execution of said code block.
15. (Previously Presented) The apparatus of claim 9, wherein the processor is able to, before insertion, translate said code block from said first format to said second format.
16. (Original) The apparatus of claim 9, wherein the first computing platform is a 32-bit based computing platform and the second computer architecture is a 64-bit based computing platform.
17. (Previously presented) A computing platform comprising:
a hardware processor to insert one or more instructions, during translation of a code block from a first format suitable for a first computing platform to a second format suitable for a second computing platform, in said code block to detect whether

execution of said code block results in misaligned data access prior to execution of said code block; to store a location of a first memory address in a tracking list for storing the location of memory addresses for which misaligned data access is detected if accessing the first memory address by a first instruction results in the misaligned data access; to add one or more instructions to the first instruction; to check the tracking list to determine if a location of a second memory address is identical to the location of the first memory address when a second instruction requires access to the second memory address; and to obviate the need to add instructions to the second instruction if the location of the second memory address accessed by the second instruction is identical to the location of the first memory address; and a dynamic random access memory operably associated with said processor to store at least a portion of said code block.

18. (Cancelled)
19. (Cancelled)
20. (currently amended) The ~~apparatus~~ computing platform of claim 17, wherein the processor is able to insert at least one instruction in said code block to detect a location of an instruction whose execution results in the misaligned data access.
21. (Previously presented) A machine-readable medium having stored thereon a set of instructions that, if executed by a machine, cause the machine to perform a method comprising:
 - during translation of a code block from a first format suitable for a first computing platform to a second format suitable for a second computing platform, inserting one or more instructions in said code block to detect whether execution of said code block results in the misaligned data access prior to execution of said code block;

storing a location of a first memory address in a tracking list for storing the location of memory addresses for which misaligned data is detected if accessing the first memory address by a first instruction results in the misaligned data access;

adding one or more instructions to the first instruction;

checking the tracking list to determine if a location of a second memory address is identical to the location of the first memory address when a second instruction requires access to the second memory address; and

obviating the need to add instructions to the second instruction if the location of the second memory address accessed by the second instruction is identical to the location of the first memory address.

22. (Cancelled)
23. (Previously Presented) The machine-readable medium of claim 21 wherein the instructions that result in detecting result in insertion of at least one instruction in said code block to detect a location of an instruction whose execution results in the misaligned data access.
24. (currently amended) The machine-readable medium of claim 21, wherein the instructions that result in insertion resulting insertion of at least one instruction in said code block to detect a location of an instruction whose execution results in the misaligned data access.
25. (Original) The machine-readable medium of claim 21, wherein the instructions comprise at least part of a translator.
26. (Original) The machine-readable medium of claim 21, wherein the instructions comprise at least part of an execution layer.
27. (Original) The machine-readable medium of claim 21, wherein the instructions comprise at least part of an operating system.

28. (Original) The machine-readable medium of claim 21, wherein the instructions comprise at least part of a compiler.
29. (Previously Presented) The method of claim 1, wherein the location of the second memory address is identical to the location of the first memory address if the size of the difference between the first and second memory addresses is the same as the size of the data misalignment or is a factor of the size of the data misalignment.
30. (Previously Presented) The apparatus of claim 9, wherein the location of the second memory address is identical to the location of the first memory address if the size of the difference between the first and second memory addresses is the same as the size of the data misalignment or is a factor of the size of the data misalignment.
31. (Previously Presented) The apparatus of claim 17, wherein the location of the second memory address is identical to the location of the first memory address if the size of the difference between the first and second memory addresses is the same as the size of the data misalignment or is a factor of the size of the data misalignment.
32. (Previously Presented) The machine-readable medium of claim 21, wherein the location of the second memory address is identical to the location of the first memory address if the size of the difference between the first and second memory addresses is the same as the size of the data misalignment or is a factor of the size of the data misalignment.